

008-0583

A Comparison of Pull Control Policies in Hybrid Production Systems

Aybek Korugan, Özge Çadırcı

Boğaziçi University, Department of Industrial Engineering, 34342 Bebek, Istanbul, Turkey

E-mail: aybek.korugan@boun.edu.tr Phone: +90 212 359 75 21

POMS 19th Annual Conference

La Jolla, California, U.S.A.

May 9 to May 12, 2008

A Comparison of Pull Control Policies in Hybrid Production Systems

Aybek Korugan, Özge Çadircı

Boğaziçi University, Department of Industrial Engineering, 34342 Bebek, Istanbul, Turkey

ABSTRACT

This paper concentrates on a hybrid production system that performs both remanufacturing and manufacturing activities under pull type production control. The incoming demand is satisfied by the output of either process, where remanufactured products are assumed to be restored into “as good as new” condition. In the study, four of the most common pull control systems, viz. Kanban Control System (KCS), Base Stock Control System (BSCS), Generalized Kanban Control System (GKCS) and Extended Kanban Control System (EKCS), are compared on the single-stage hybrid production system considered. A stochastic model is developed for each of the four control policies. Then these models are analyzed using approximation techniques in order to obtain performance measures of interest. Using these performance measures, a cost function is defined. This cost function is then minimized with respect to the control parameters of each control mechanism. Finally, results derived from numerical experimentation are obtained, and conclusions are drawn.

1. Introduction:

Environmental issues have been forcing companies to reuse old products increasingly. Reuse activities can be classified as recycling, remanufacturing and direct reuse. Remanufacturing is the collection of activities whose aim is to bring a used product back to its useful state (Korugan and Gupta, 2001). When this profitable and environmentally friendly

process, i.e. remanufacturing, is combined with a manufacturing process, this whole system is called a hybrid production system. In this paper we concentrate on companies that are involved in both remanufacturing and manufacturing activities simultaneously, viz. a hybrid production system is investigated.

Much work has been carried out on control policies such as kanban, base stock, generalized kanban and extended kanban, but relatively few comparison studies have been conducted (Duri *et al.*, 2000b). Moreover, these studies are based on a manufacturing system, namely a hybrid remanufacturing / manufacturing system was not considered at all. Besides, an infinite number of raw parts were always supplied for the production system, however, any return flow according to a specific distribution has not been considered and integrated with the production system before.

We focus on material flow control systems in which production is triggered by actual demands that are often referred to as pull control systems. The aim of this paper is to compare performances of kanban, base stock, generalized kanban and extended kanban control systems considered in a hybrid remanufacturing / manufacturing system environment using continuous-time Markov chains (CTMCs). After modeling these control systems as queuing networks with synchronization mechanisms, we obtain steady-state probabilities for each state by expressing the behavior of the queuing network as CTMCs which will then be used for calculating performance measures of interest. Using these performance measures we obtain cost functions for each system and compare them while changing the values of parameters.

The paper is organized as follows: Section 2 provides a description of each control policy we consider. Section 3 gives a brief presentation of the analytical methods used to get the performance measures. Section 4 encompasses cost optimization part of our study. Numerical results are reported in Section 5. Finally, conclusions are drawn in Section 6.

2. Presentation of pull control policies in hybrid production environment

2.1. General remarks

In our hybrid production system consisting of remanufacturing and manufacturing processes and stocking points, remanufacturable products enter the remanufacturable inventory if the current queue length of core buffer is convenient, otherwise they are disposed of. The items available in remanufacturable inventory are to be remanufactured and to be put into the serviceable inventory, however, the output of the remanufacturing process may be insufficient to fulfill all the incoming demands. Therefore, a manufacturing process exists to produce new modules (Van der Laan *et al.*, 1999). In the remainder of the paper “cores” will be used instead of “returned items” as a matter of convenience.

Each system can be modeled as a queuing network with synchronization mechanisms. Furthermore, the following assumptions hold for each control system, viz., we limit our study to a hybrid production system composed of a remanufacturing process and a manufacturing process and producing only one type of product. We also assume that the external demands are backordered when no finished product is present in the output buffer. Manufacturing process does not need any demand queue in front of it since sufficient raw parts are always supplied whenever needed, however, cores arrive at the system according to Poisson process, i.e. return flow is Poisson distributed with return rate γ . The service-time distribution of each server is represented by an exponential distribution with service-rates of μ_1 and μ_2 for remanufacturing machine and for manufacturing machine, respectively.

The triggering procedure where the predefined probability mentioned above, viz. routing probability (r), remains constant is called “static routing mechanism” (Korugan and Gupta, 2001b). The arrival process of external demands is Poisson with a demand rate of λ_D . The buffer for cores (P_0) has a finite capacity of size B . This means that if a core arrives at the system as the current queue length of the cores is B , then this core is disposed of.

2.2. Kanban Control System (KCS)

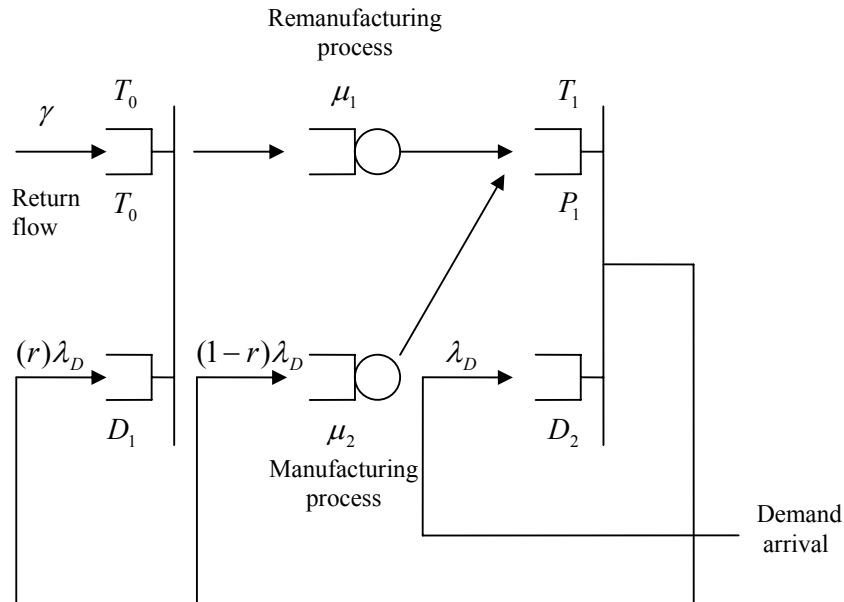


Figure 1. Kanban controlled hybrid production system

The first synchronization station T_0 represents the synchronization between a core and a free kanban in the hybrid production system whereas the second synchronization station T_1 represents the synchronization between a finished product and an external demand.

Upon the arrival of a customer demand, this demand joins the backorder queue (D_2), thereby requesting the release of a finished product from the output buffer (P_1) down to the customer. If a finished product is available in the output buffer, it is released to the customer after liberating the kanban that was attached to it. This kanban is transferred upstream either to the manufacturing process with probability $(1-r)$, thereby requesting a finished product via manufacturing process or to the demand queue for cores (D_1) with probability (r) carrying along with it a demand for obtaining a finished part via remanufacturing and authorizing the release of a core into the remanufacturing process. As it is obvious, the demand queue of

cores contains pairs of demands and remanufacturing authorizations. The hybrid production system has a fixed number of kanbans represented as K .

2.3. Base Stock Control System (BSCS)

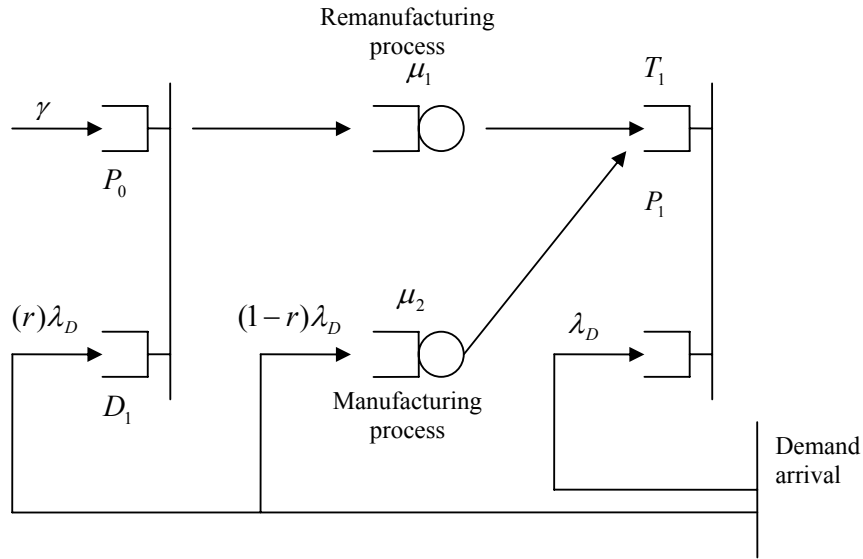


Figure 2. Base stock controlled hybrid production system

BSCS applied to a hybrid production system depends on only one parameter per stage that determines the target in terms of the number of products to be produced and stored at the output of the whole system. The purpose of BSCS is to satisfy demands and to lead P_1 to its maximum level S .

BSCS operates as follows. Upon the arrival of a customer demand, this demand joins the backorder queue (D_2), thereby requesting the release of a finished product from the output buffer (P_1) down to the customer. If a finished product is available in the output buffer, it is directly released to the customer. The customer demand instantly also generates a demand in either D_1 that authorizes the release of a core from P_0 to the remanufacturing process with probability (r) or this demand can also be generated for the manufacturing process with probability ($1-r$). Demand generation behavior of the BSCS leads to an immediate start either

for working on a core which it pulls from the buffer of cores or for working on a new raw part to be manufactured, no matter whether it is necessary or not. Therefore, it becomes urgent that WIP levels grow up very rapidly without any limit (Liberopoulos and Dallery, 2000).

2.4. Generalized Kanban Control System (GKCS)

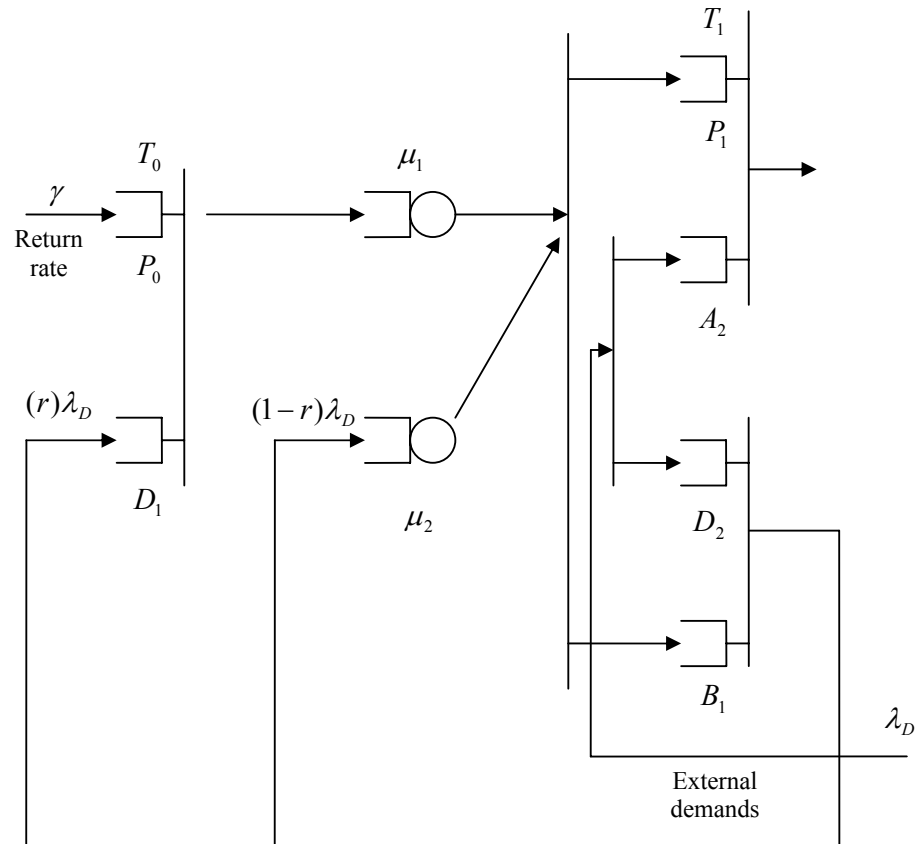


Figure 3. Generalized kanban controlled hybrid production system

GKCS gets the idea of safety stocks from the BSCS and production authorization cards from the KCS, so that GKCS is defined by two parameters per stage, one defining the safety stocks (S) and the other the number of production authorization cards (K) (Karaesmen and Dallery, 2000).

The behavior of GKCS is as follows. As soon as there is one core in queue P_0 and one kanban in queue D_1 , a core / kanban pair is transferred to the remanufacturing process where it will be treated. Again- since after a kanban is freed, it returns to either to queue D_1 or to the

manufacturing process due to static routing mechanism, and since sufficient raw parts are always available for being manufactured- as soon as there is one kanban available for manufacturing, a raw part / kanban pair is transferred to the manufacturing process where it will be treated. When either process is complete, the kanban is separated from the product to join queue B_1 and the product enters into queue P_1 (Duri *et al.*, 2000b). Upon the arrival of an external demand, this demand is split into two entities. One of these entities joins queue A_2 and enables the consumption of a finished product from queue P_1 . The other one joins queue D_2 in order to be transferred upstream.

2.5. Extended Kanban Control System (EKCS)

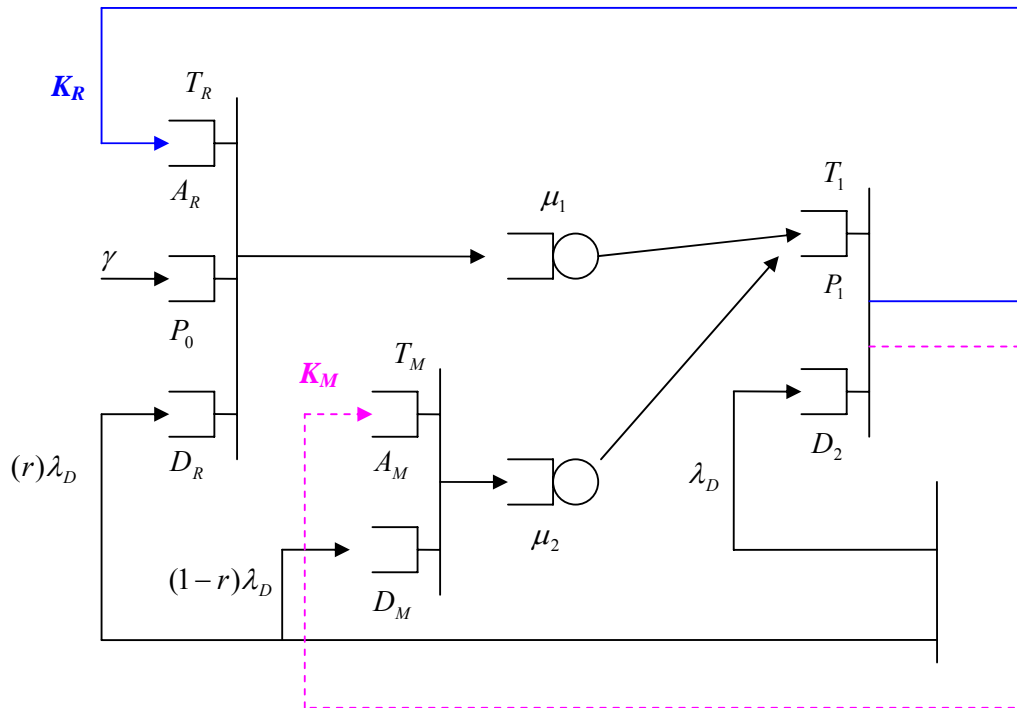


Figure 4. Extended kanban controlled hybrid production system

Extended Kanban Control System (EKCS) can be regarded as an integration of conventional KCS and BSCS in the sense that it is defined by two parameters per stage,

namely the number of kanbans, K , and the base stock of parts in inventory, S . EKCS seems to be similar to GKCS in terms of its multi-parameter structure. However, it is an extension of GKCS. EKCS is a control system easier to apply in comparison to GKCS. Furthermore, parts in an EKCS are easier to trace as a result of having kanbans attached to them, whereas in the GKCS finished parts are separated from their kanbans.

To control our hybrid production system with an EKCS we have kanbans dedicated for both remanufacturing and manufacturing processes that flow throughout the system. However, the demand triggering procedure is static routing, i.e. demands return with a predefined probability either to remanufacturing or manufacturing process.

The behavior of the system is as follows. Upon the arrival of an external demand, a finished product from queue P_1 is consumed if there is any. Otherwise, this demand waits in queue D_2 , in other words, it is backordered until a finished product becomes available. The arrival of an external demand also generates a demand in D_R with probability (r) or in D_M with probability ($1-r$), as is the case in the BSCS. A core is not immediately released from P_0 into the remanufacturing process unless there is a free kanban in A_R . The same logic is valid for manufacturing process, too, that is, if an external demand generates a demand in D_M with probability ($1-r$), a free kanban must be available in A_M in order a raw part to enter the manufacturing process. In short, we can say, that to enter any process three entities are needed, a part to be processed, a free kanban to be attached and a demand for the part. After being processed with a service rate of either μ_1 or μ_2 by the remanufacturing machine and the manufacturing machine, respectively, the finished products are stored in the output buffer P_1 where they await removal to be consumed by the customers.

3. Analytical methods for performance evaluation

3.1. Kanban Control System (KCS)

We consider our single-stage hybrid production system as a queuing network with synchronization stations. This system consists of four stations totally, i.e. two synchronization stations at the input and output, one station for remanufacturing and one station for manufacturing. Let K be the total number of customers (kanbans) of our one-stage hybrid production system. The set of indexes of the stations that are visited by class- r customers where $r = 1$, is represented by $S(1) = \{1,2,3,4\}$. The idea of this method is to associate with the original network, R single-class product form networks which represent the number of classes (stages), where R becomes 1 in our case. For each station i of the original network visited by its customers, we associate a load-dependent exponential service station, as depicted in Figure 5. Let $\mu_i(n_i)$, $n_i = 1, \dots, K$, $i = 1, 2, 3, 4$ denote the load-dependent service rates of station i for the single-class product-form network. The visit ratios are simply calculated and are as follows: $V_1 = r$, $V_2 = r$, $V_3 = 1 - r$ and $V_4 = 1$. The single-class network is known as Gordon-Newell network (Baynat *et al.*, 2001), so that we have the opportunity of utilizing product-form solution to get the steady-state probabilities $P(n)$ for the network. The product-form solution is obtained by means of the below formula (Formula 1). G is the normalization constant associated with the one-stage network. The normalization constant approach can be found in Bruell and Balbo's study. In a closed queuing network like KCS, a fixed number of customers (K) circulate through the network at all times. Here, the customers have four stations to be visited. So, the state of such a network can be described by a vector $n = (n_1, n_2, n_3, n_4)$ where n_i represents the current number of customers present at the i th facility ($\sum_{i=1}^4 n_i = K$).

$$P(n_1, n_2, n_3, n_4) = \frac{1}{G} \prod_{i \in S(1)} \left[\prod_{n=1}^{n_i} \frac{V_i}{\mu_i(n)} \right] \quad (1)$$

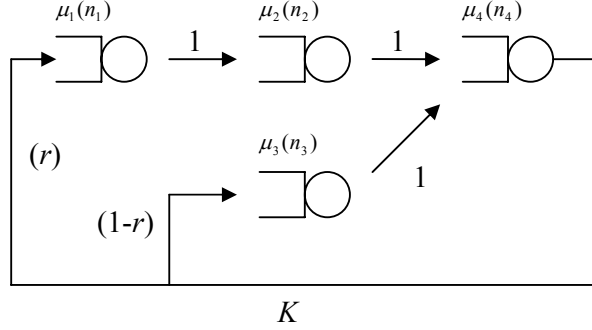


Figure 5. One-class product-form network

The first level approximation of the method happens where the performance of the single class of customers in the original network is approximated by the performance of the single class product-form network. It remains to specify the load-dependent service rates $\mu_i(n_i)$, $n_i = 1, \dots, K$, $i = 1, 2, 3, 4$. These service rates of the flow equivalent service centers of the product-form network should be equal to the conditional throughputs $v_i(n_i)$ of class-1 customers (due to one-staged system) at the corresponding station in the original network. Since exact values of the conditional throughputs require an exact solution of the original system, a second level of approximation is needed where conditional throughputs are approximated. In order to approximate the conditional throughputs of our one-class at every station in the original system, the stations are analyzed in isolation.

Each station i is analyzed as a queue fed by external arrival processes, as depicted in Figure 6. The arrival process of customers of our one-class is assumed to be a state-dependent Markovian process with rates $\lambda_i(n_i)$, $n_i = 0, \dots, K-1$, where n_i is the current number of customers present at station i .

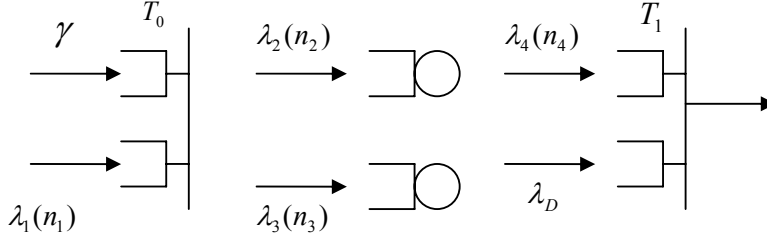


Figure 6. Decomposition of the network to the isolated stations with load dependent arrival rates

We first suppose that $\lambda_i(n_i)$ are given, such that $i \in S(1)$ and $n_i = 0, \dots, K-1$. Station i can be analyzed in isolation using continuous-time Markov chains (CTMCs). The underlying Markov chain depicted below (Figure 7) represents the behavior of the first synchronization station T_0 . This synchronization station is fed by two Markovian arrival processes with state-dependent arrival rates on one hand, and with external resources on the other. The state of this CTMC is (n_1, n_r) , where n_1 is the current number of kanbans and n_r is the current number of cores. The steady-state probabilities can be obtained by means of the balance equations, and marginal probabilities can then be derived as follows:

$$\tilde{P}_1(n_1) = \frac{\prod_{i=0}^{n_1-1} \lambda_1(i)}{\gamma^{n_1}} p(0,0) \quad \text{for } n_1 = 1, \dots, K \quad (2)$$

$$\tilde{P}_1(0) = \frac{\left(\frac{\gamma}{\lambda_1(0)}\right)^{B+1} - 1}{\frac{\gamma}{\lambda_1(0)} - 1} p(0,0) \quad (3)$$

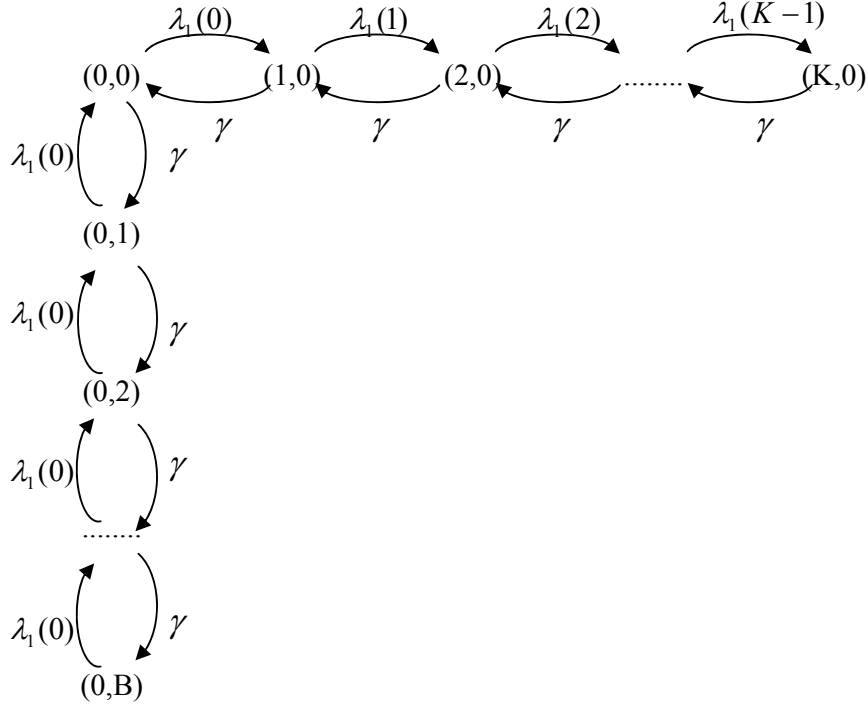


Figure 7. Markov chain representing the behavior of the first synchronization station T_0

The iterative method uses these marginal probabilities to estimate the conditional throughputs $\tilde{v}_i(n_i)$ such that:

$$\tilde{v}_i(n_i) = \lambda_i(n_i - 1) \frac{\tilde{P}_i(n_i - 1)}{\tilde{P}_i(n_i)} \quad \text{for } n_i = 1, \dots, K \quad (4)$$

The load-dependent service rates of the associated stations in the single-class product-form network are set equal to these estimated conditional throughputs, i.e.:

$$\mu_i(n_i) = \tilde{v}_i(n_i) \quad \text{such that } i \in S(1) \text{ and } n_i = 1, \dots, K \quad (5)$$

The state-dependent arrival rates $\lambda_i(n_i)$ of one-class customers at the different stations are required in order to use this approach. These quantities can be obtained from the product-form solutions of the single-class product-form network given by Formula 1.

$$\lambda_i(n_i) = \mu_i(n_i + 1) \frac{P_i(n_i + 1)}{P_i(n_i)} \quad \text{for } n_i = 0, \dots, K - 1 \quad (6)$$

To sum up the whole procedure, marginal probabilities obtained by product-form solution and normalization constant approach are used to obtain state-dependent arrival rates, which are then utilized to estimate conditional throughputs after deriving the marginal probabilities from the solution of CTMCs. The load-dependent service rates are set equal to these estimated conditional throughputs next. The whole story is repeated until a required level of tolerance is captured in convergence of the load-dependent service rates.

After running the algorithm and obtaining convergence, the necessary performance parameters such as the average queue length of cores, the average queue length of finished products, the average queue length of backorders, average work in process can be calculated by means of expected values.

3.2. Generalized Kanban Control System (GKCS)

In order to approximate the conditional throughputs of our one-class at every station in the original system, the stations are analyzed in isolation. Only the last synchronization station has a different structure, all the other stations including the first synchronization station and both processes behave the same as in the KCS. Therefore, only the last synchronization station is depicted below in the context of isolated analysis. After the convergence of the analytical method we are informed about the load-dependent arrival rates of kanbans in queue B_1 denoted by $\lambda(n_B)$. The last synchronization station is made up of two synchronization stations which contain two queues, namely, P_1 and A_2 on the one hand, D_2 and B_1 on the other.

In order to get the performance measures, we must evaluate $P(n_B = n)$ which corresponds to the probability of having n free kanbans. To model the behavior of the station represented in Figure 9 the corresponding Markov chain is constructed (Figure 8). The state vector is chosen as (n_B, n_D, n_A, n_P) , whose dimensions represent the current number in the free kanban queue, in the demand queue, in the backorder queue and finished product queue,

respectively. Considering the values of K and S , two cases must be distinguished for the construction of the Markov chain (Duri *et al.*, 2000b).

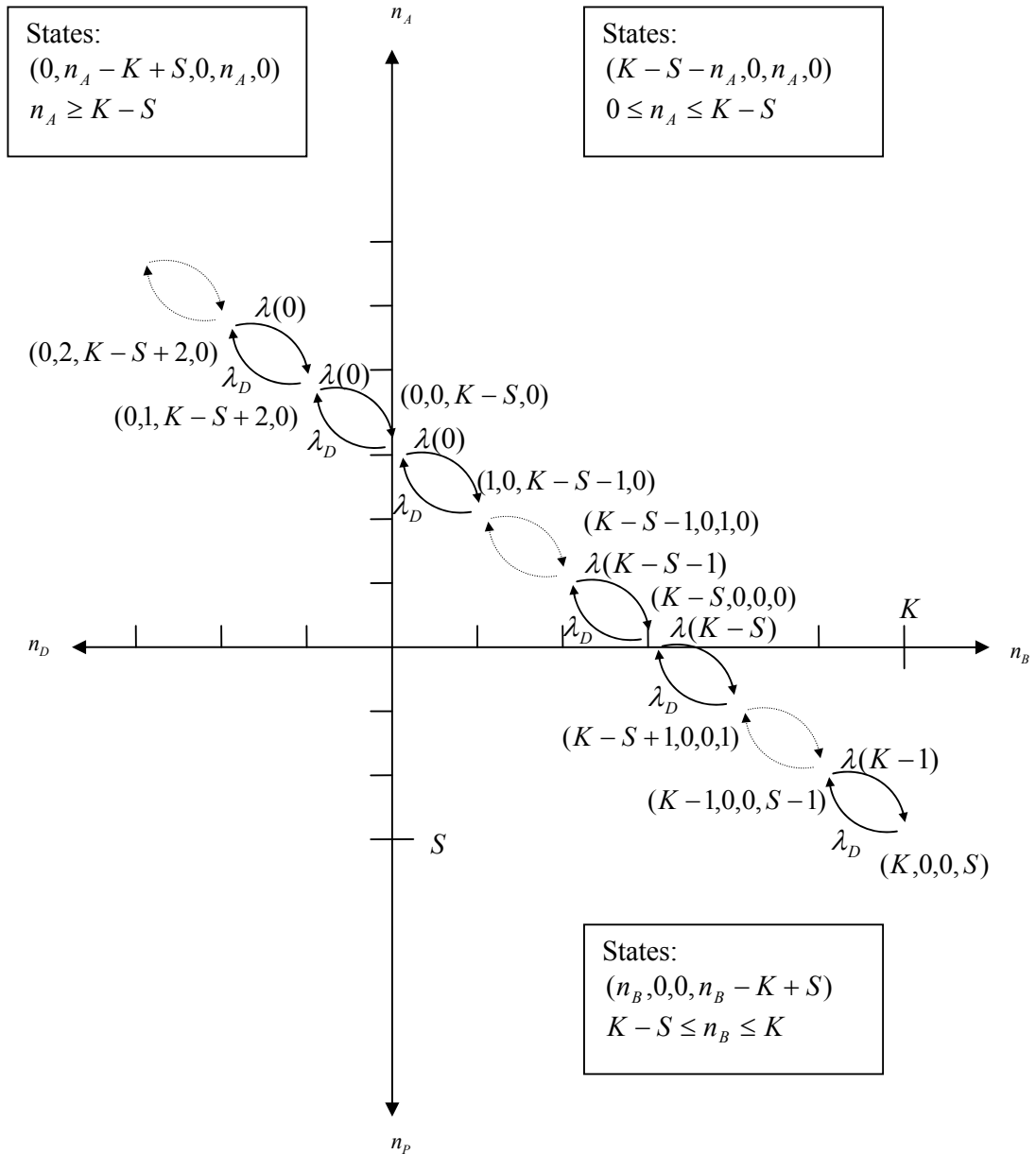


Figure 8. Markov chain associated with the last synchronization station in case $K \geq S$

(adapted from Duri *et al.*, 2000b)

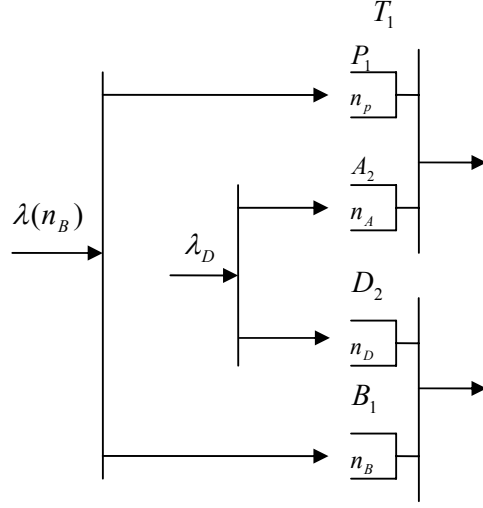


Figure 9. Isolated version of the last synchronization station with load dependent arrival rates

Just as for KCS, after writing the balance equations and all the states of the Markov chain as functions of $P(0,0,K-S,0)$ and using the property that the sum of the probabilities of all possible states must be equal to one, we deduce:

$$P(0,0,K-S,0) = \left(\frac{1}{1 - \frac{\lambda_D}{\lambda(0)}} + \sum_{n=1}^K \frac{\prod_{i=0}^{n-1} \lambda(i)}{(\lambda_D)^n} \right)^{-1} \quad (7)$$

$$\tilde{P}(n_B = n) = \frac{\prod_{i=0}^{n-1} \lambda(i)}{(\lambda_D)^n} P(0,0,K-S,0) \quad \text{for } n = 1, \dots, K \quad (8)$$

$$\tilde{P}(n_B = 0) = \frac{1}{1 - \frac{\lambda_D}{\lambda(0)}} P(0,0,K-S,0) \quad (9)$$

For calculating the average backorder queue length, the probability $P(n_p = 0)$ is needed, and for calculating the average finished product queue the probability $P(n_p = n)$ is needed. Therefore, the following probabilities must be calculated in addition.

$$\tilde{P}(n_p = n) = \frac{\prod_{i=0}^{K-S+n-1} \lambda(i)}{(\lambda_D)^{K-S+n}} P(0,0,K-S,0) \quad \text{for } n=1,\dots,S \quad (10)$$

$$\tilde{P}(n_p = 0) = 1 - \sum_{n=1}^S P(n_p = n) \quad (11)$$

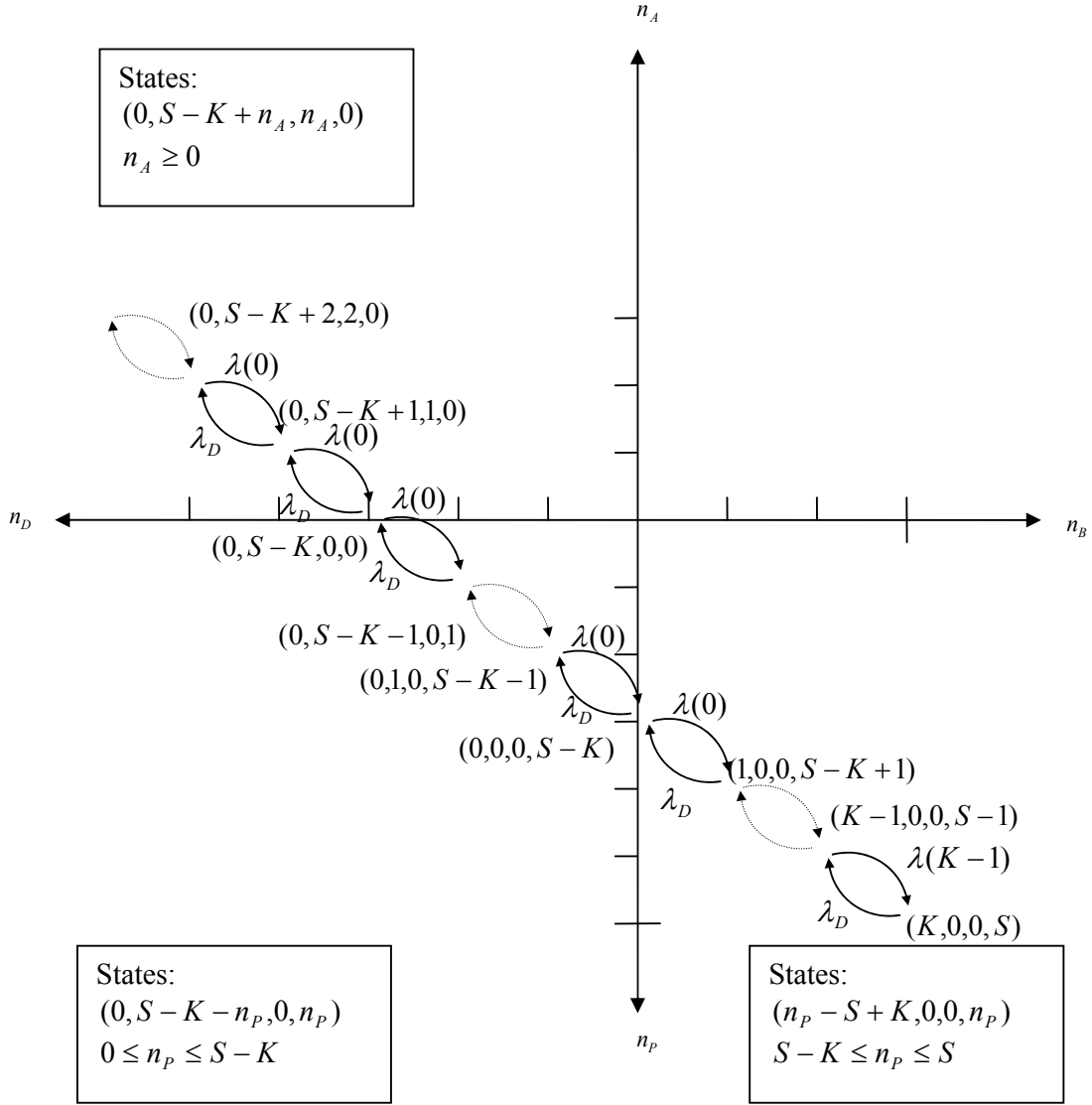


Figure 10. Markov chain associated with the last synchronization station in case $K < S$

(adapted from Duri *et al.*, 2000b)

For the second case where $K < S$ shown in Figure 10, marginal probabilities can be obtained as follows:

$$P(0, S - K, 0, 0) = \frac{\left(\frac{\lambda_D}{\lambda(0)}\right)^{S-K}}{\frac{1}{1 - \frac{\lambda_D}{\lambda(0)}} + \sum_{n=1}^K \frac{\prod_{i=0}^{n-1} \lambda(i)}{(\lambda_D)^n}} \quad (12)$$

$$\tilde{P}(n_B = n) = \frac{\prod_{i=0}^{n-1} \lambda(i)}{(\lambda_D)^n} \left(\frac{\lambda(0)}{\lambda_D}\right)^{S-K} P(0, S - K, 0, 0) \quad \text{for } n = 1, \dots, K \quad (13)$$

$$\tilde{P}(n_B = 0) = \left(\sum_{j=1}^{S-K} \left(\frac{\lambda(0)}{\lambda_D}\right)^j + \frac{1}{\frac{\lambda(0)}{\lambda_D} - 1} \right) P(0, S - K, 0, 0) \quad (14)$$

The following probabilities must be calculated, too.

$$\tilde{P}(n_p = n) = \left(\frac{\lambda(0)}{\lambda_D}\right)^n P(0, S - K, 0, 0) \quad \text{for } n = 1, \dots, S - K \quad (15)$$

$$\tilde{P}(n_p = n) = \frac{\prod_{i=0}^{n-S+K-1} \lambda(i)}{(\lambda_D)^{n-S+K}} \left(\frac{\lambda(0)}{\lambda_D}\right)^{S-K} P(0, S - K, 0, 0) \quad \text{for } n = S - K + 1, \dots, S \quad (16)$$

$$\tilde{P}(n_p = 0) = 1 - \sum_{n=1}^S P(n_p = n) \quad (17)$$

3.3. Extended Kanban Control System (EKCS)

We convert the EKCS into a closed-loop network so that a product-form approximation technique can be applied to get the performance parameters (Figure 11). The content of queue $S_R + S_M$ is finished products with dedicated kanbans attached to them. The content of queue

F_R is free remanufacturing kanbans and queue F_M contains free manufacturing kanbans. The queues below these queues represent the demand queues. If we suppose the kanbans attached to the finished products to be detached and add them to the queues concerned, it turns out, that the maximum number in the queues of free kanbans increase from F_R and F_M to K_R and K_M , respectively. Figure 12 represents the last synchronization station in isolation with load-dependent arrival rates. n_{BR} and n_{BM} are denoted as the current number in remanufacturing and manufacturing kanban queue, respectively.

Let n_R be a variable that represents n_{BR} when it has positive value, and n_{DR} when it has negative value. Also, let n_M be a variable that represents n_{BM} when it has positive value, and n_{DM} when it has negative value. Moreover, let n_{FP} be a variable that represents n_p when it has positive value, and n_d when it has negative value. Then, if we follow the movements that occur in the queues, it can be seen that $n_{FP} = S - [(K_R - n_R) + (K_M - n_M)]$. $(K_R - n_R)$ represents the number of arrived remanufacturing demands, whereas $(K_M - n_M)$ represents the number of arrived manufacturing demands. The aim of finding out such an expression is to relate the current number in the finished product queue with the current number in the remanufacturing and manufacturing kanban queues. For example let K_R and K_M be equal to three, respectively. It turns out, that the above formula becomes $n_{FP} = (S - 6) + n_R + n_M$. Hence, we can draw a picture of the situation, as depicted in Figure 13.

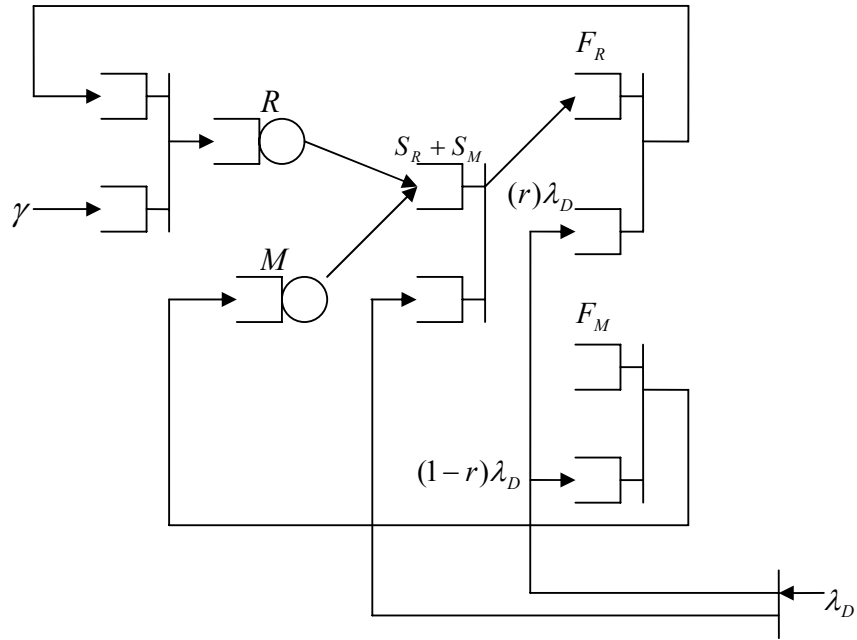


Figure 11. EKCS converted to a closed network

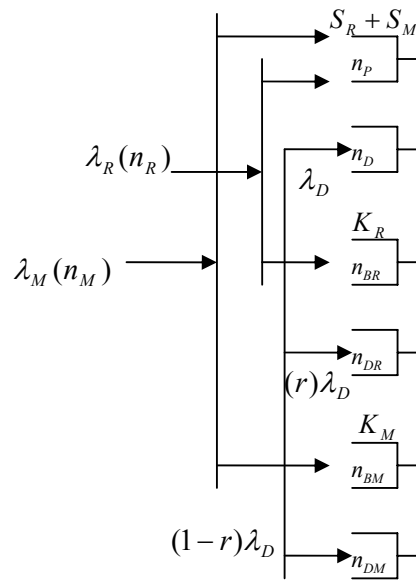


Figure 12. Last synchronization station in isolation

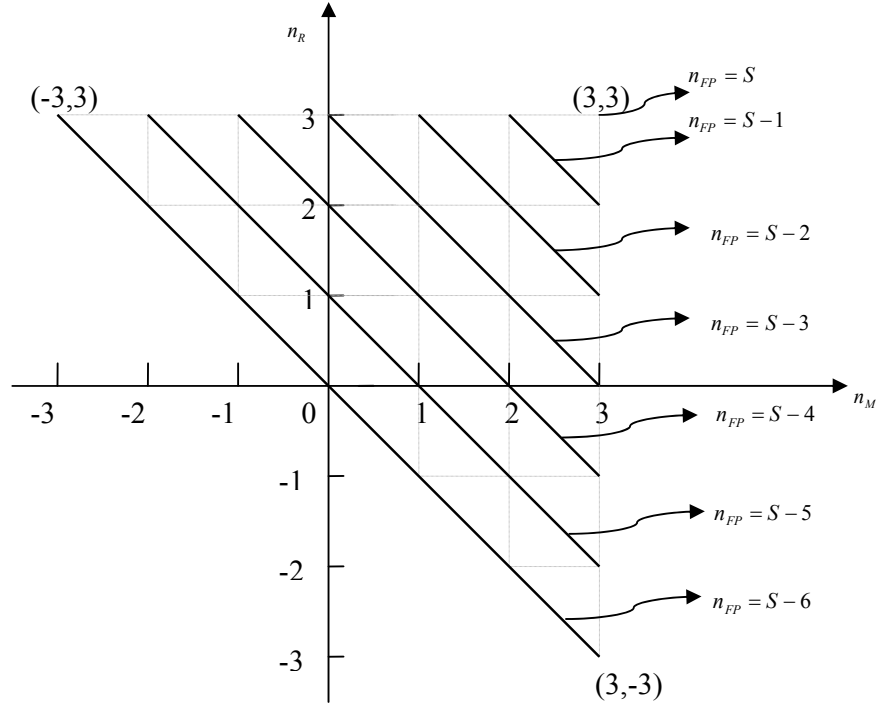


Figure 13. An example for $K_R = K_M = 3$

Using the relations in Figure 13, it is obvious that:

$$P(n_{FP} = S - i) = \sum_{j=0}^i P(n_R = K_R - i + j, n_M = K_M - j) \quad (18)$$

Since we assume that n_R and n_M are independent, it turns out that:

$$P(n_{FP} = S - i) \approx \sum_{j=0}^i P(n_R = K_R - i + j) P(n_M = K_M - j) \quad (19)$$

These probabilities are utilized to calculate the average finished product queue and the backorder queue. The calculation of the probabilities needs solving Markov chains for states (n_{BR}, n_{DR}) and (n_{BM}, n_{DM}) where the first dimensions represent the current number in kanban queues for remanufacturing and manufacturing, respectively, and the second dimensions represent the current number of demand (backorder) queues for them. Below is

the Markov chain of state (n_{BR}, n_{DR}) depicted (Figure 14). After that, the analysis is given by means of marginal probabilities.

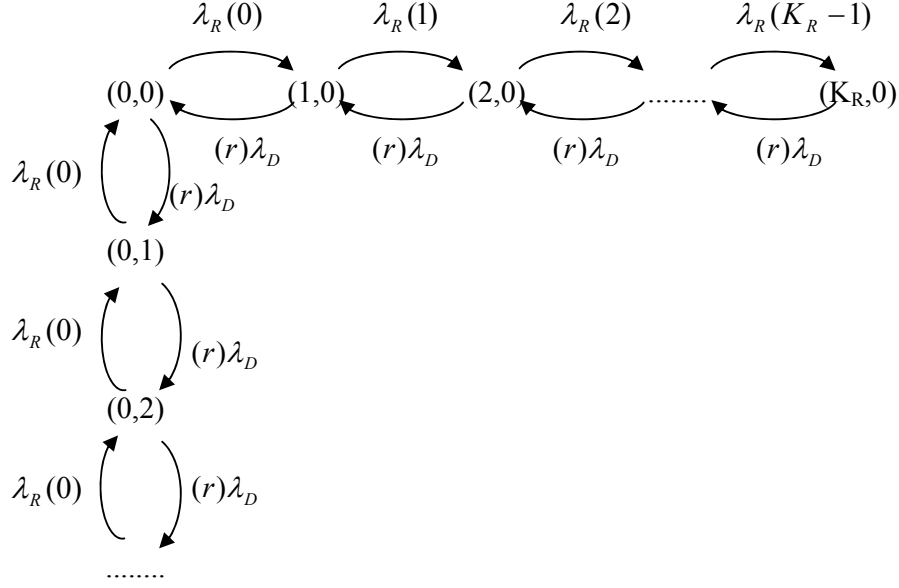


Figure 14. Markov chain representing the behavior of the state (n_{BR}, n_{DR})

$$P(n_R = i) = \frac{\prod_{n=0}^{i-1} \lambda_R(n)}{(r\lambda_D)^i} P(n_R = 0) \quad \text{for } i = 1, \dots, K_R \quad (20)$$

$$P(n_R = -i) = \left(\frac{r\lambda_D}{\lambda_R(0)} \right)^i P(n_R = 0) \quad \text{for } i = 1, \dots, \infty \quad (21)$$

$$P(n_R = 0) = \left(\frac{1}{1 - \frac{r\lambda_D}{\lambda_R(0)} + \sum_{i=1}^{K_R} \frac{\prod_{n=0}^{i-1} \lambda_R(n)}{(r\lambda_D)^i}} \right)^{-1} \quad (22)$$

$$\tilde{P}(n_{BR} = i) = P(n_R = i) \quad \text{for } i = 1, \dots, K_R \quad (23)$$

$$\tilde{P}(n_{BR} = 0) = \sum_{i=0}^{\infty} P(n_R = -i) \quad (24)$$

3.4. Base Stock Control System (BSCS)

In (Dallery and Liberopoulos, 2000) it is pointed out that the EKCS with $K_i = \infty$, $S_i \geq 0, i = 1, \dots, N$, is equivalent to the BSCS having a base stock of S_i finished parts in stage i , according to property eight.

In order to be able to solve our BSCS applied to a hybrid production environment analytically, we use this beneficial property. We make use of the same algorithm that we used for EKCS where we set the total kanban size equal to infinity.

The indicator, that points us out the similarity between the EKCS and BSCS is the value of probabilities $P(n_{BR} = 0)$ and $P(n_{BM} = 0)$. We want the demand to immediately enter the system in the EKCS so that it can resemble BSCS. This is possible, in case that there are kanbans available in the system. Without any kanbans the demand is not allowed to be transmitted. Hence, the idea is that, the more kanbans the EKCS has, the closer it gets to BSCS.

4. Optimization

The comparison is made in terms of minimum costs achieved for each control policy under a given parameter set. The total cost function consists of four components, viz. holding cost, backorder cost, production cost and disposal cost.

The cost parameters are calculated according to Teunter *et al.*, (2000). Here, the fifth method for setting the holding cost rates in an average cost model outperforms other methods. So we make use of the fifth method.

Our models consider three types of stocked items, namely, non-serviceable items- these are cores that are not yet remanufactured-, remanufactured items and manufactured items. Thus, holding cost is easily calculable using average queue lengths and holding cost rates. The method considers the cost for collecting cores as fixed. Therefore, this cost is not included in the opportunity cost rate. Moreover, the cost associated with transporting a

returned non-serviceable item to either its stocking location or to the disposal facility is ignored. In our models, both types of serviceable items can be in stock simultaneously. However, the method that we use eliminates the need for a stock depletion rule by using the same holding cost rate for all serviceable items. The necessary notations are as follows:

Table 1. The necessary notations for cost structure

λ_D	external demand rate
γ	return rate
r	static routing probability
α	inventory carrying charge per month
h	unit out-of-pocket holding cost rate per month
c_M	marginal cost for manufacturing one item
c_R	marginal cost for remanufacturing one item
c_D	cost for disposing one item
b	unit backorder cost per month

Table 2. Underlying cost components

Holding cost for non-serviceable items = $[h + \alpha(c_M - c_R)][QR + WIPR + WIPM]$
Holding cost for serviceable items = $[h + \alpha c_M][QFG]$
Backorder cost = $b[QD]$
Disposal cost = $c_D[\gamma - r\lambda_D]$
Production cost for remanufacturing = $c_R[TH_R]$
Production cost for manufacturing = $c_M[TH_M]$

The components of the total cost function are given in Table 2. Here, QR stands for the average queue length of cores. $WIPR$ and $WIPM$ represent the average work-in-process of remanufacturing and manufacturing (non-serviceable items), respectively. QFG stands for the average queue length of finished (serviceable items) goods. QD represents the average backorder queue. TH_R and TH_M imply expected throughputs of remanufacturing and

manufacturing processes, respectively. Summation of the above cost components becomes our total cost function as given below.

$$Z = [h + \alpha(c_M - c_R)][QR + WIPR + WIPM] + [h + \alpha c_M][QFG] + b[QD] + c_D[\gamma - r\lambda_D] + c_R[TH_R] + c_M[TH_M] \quad (25)$$

Our aim is to minimize the total cost with respect to the control parameters related to each control policy. We search for the configuration which has the minimum cost.

To find the optimal configuration, an enumerative method is utilized that is proposed by Duri *et al.*, (2000b). All possible configurations are tested in the order of given by the algorithm. In the KCS, for instance, considering the total cost as a function of the kanban size and the static routing probability, we initialize these parameters, then we increase them by one and 0.05, respectively. This incrementation goes on until the parameters get their upper limits. The table below summarizes the underlying algorithms associated with each control policy.

Table 3. Algorithms

<p><u>Algorithm (KCS):</u> Set Z^* equal to a very large number For $K = 1$ to 20 For $r = 0.05$ to $\min(1, \gamma / \lambda_D)$ Calculate total cost Z by using the approximation method Optimality check: If $Z < Z^*$, then $Z^* = Z$, $K^* = K$ and $r^* = r$ End For End For</p>	<p><u>Algorithm 1(GKCS-1st case):</u> Set Z^* equal to a very large number For $K = 1$ to 20 For $S = 0$ to K For $r = 0.05$ to $\min(1, \gamma / \lambda_D)$ Calculate total cost Z by using the approximation method Optimality : If $Z < Z^*$, then $Z^* = Z$, $K^* = K$, $S^* = S$ and $r^* = r$ End For End For End For</p>	<p><u>Algorithm (EKCS):</u> Set Z^* equal to a very large number For $K_R = 1$ to 10 For $K_M = 1$ to 10 For $S = 0$ to $K_R + K_M$ For $r = 0.05$ to $\min(1, \gamma / \lambda_D)$ Calculate total cost Z by using the approximation method Optimality : If $Z < Z^*$, then $Z^* = Z$, $K_R^* = K_R$, $K_M^* = K_M$, $S^* = S$ and $r^* = r$ End For End For End For End For</p>
<p><u>Algorithm(BSCS):</u> Set Z^* equal to a very large number Set K_R and K_M to a very large number For $S = 0$ to 20 For $r = 0.05$ to $\min(1, \gamma / \lambda_D)$ Calculate total cost Z by using the approximation method Optimality : If $Z < Z^*$, then $Z^* = Z$, $S^* = S$ and $r^* = r$ End For End For</p>	<p><u>Algorithm 2 (GKCS-2nd case):</u> Set Z^* equal to a very large number For $K = 1$ to 20 For $S = K+1$ to 20 For $r = 0.05$ to $\min(1, \gamma / \lambda_D)$ Calculate total cost Z by using the approximation method Optimality : If $Z < Z^*$, then $Z^* = Z$, $K^* = K$, $S^* = S$ and $r^* = r$ End For End For End For</p>	

5. Numerical Analysis

5.1. Change in Backorder Cost

We conduct experiments where the return arrival rate and the demand arrival rate are set to 360 and 400 (products/month), respectively. The service rates for remanufacturing and manufacturing machines are chosen as 800 (products/month), respectively.

Figure 15 represents the per cent cost savings over BSCS as b changes and when we have $h=400$, $\alpha=0.01$, $c_R=5$, $c_M=30$, $c_D=-2.5$ and $B=10$. We observe that KCS is concave whereas EKCS is converging (Figure 15). This is because as the backorder cost increases, EKCS becomes the most beneficial policy due to its multi-parameter structure. KCS has one control parameter which is not sufficient for higher punishments. Therefore, one should avoid using simple control policies such as KCS and BSCS for higher values of backorder punishment, whereas for lower backorder punishment values, only avoiding BSCS is sufficient since the other three policies behave almost the same.

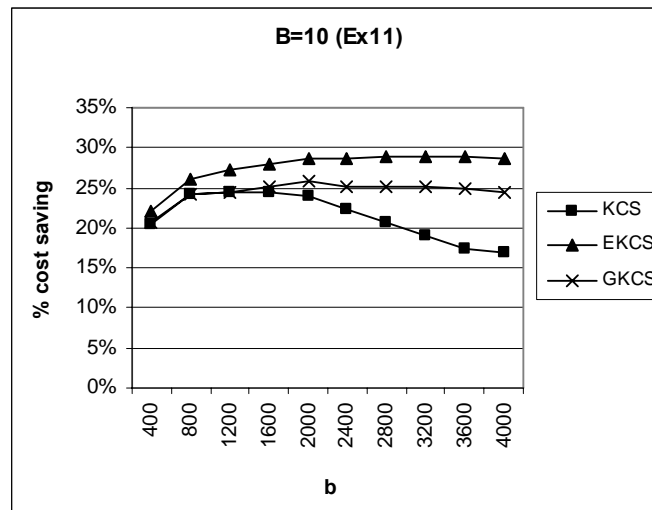


Figure 15. Per cent cost savings over BSCS as b changes

Figure 16 represents the per cent cost savings over BSCS as b changes and when we have $h=800$, $\alpha=0.01$, $c_R=10$, $c_M=30$, $c_D=2.5$ and $B=1$. For small values of b , KCS performs worse than BSCS, whereas for larger values of b , it performs better than BSCS. As b is getting larger, GKCS behaves like KCS. This follows from the fact that, when GKCS has $S=K$, it is equivalent to KCS (Frein *et al.*, 1995). However, EKCS is the leading policy when considering its increasing per cent cost savings with an increase in b (Figure 16).

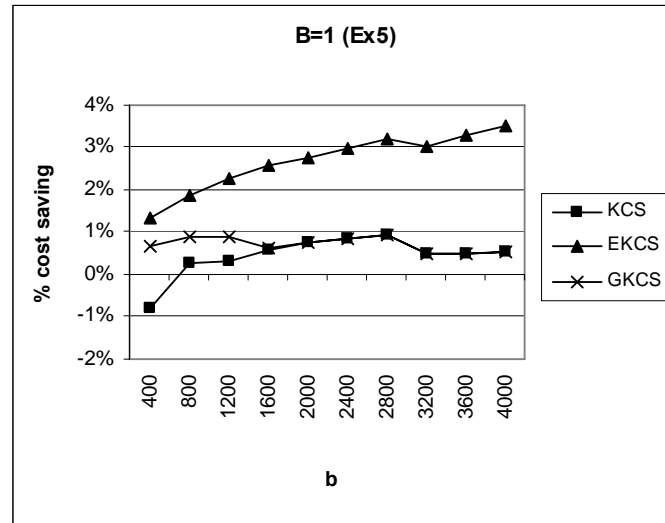


Figure 16. Per cent cost savings over BSCS as b changes

5.2. Change in Out-of-Pocket Holding Cost

Figure 17 represents the per cent cost savings over BSCS as h changes and when we have $b=3200$, $\alpha=0.01$, $c_R=5$, $c_M=30$, $c_D=-2.5$ and $B=10$. GKCS decreases the finished product queue by descending S , and, it does not change WIP values by holding K constant, because the transfer of demands from downstream to upstream can be done independently of the consumption of a finished product. This is the main reason for GKCS performing better than KCS. The multi-parameter structure of EKCS results in the lowest minimum expected total costs. As h increases, and when the buffer for cores has a high capacity, it is reasonable to prefer KCS. As the difference between the best control policy and this one is not significant, it is not worth investing in a sophisticated control system.

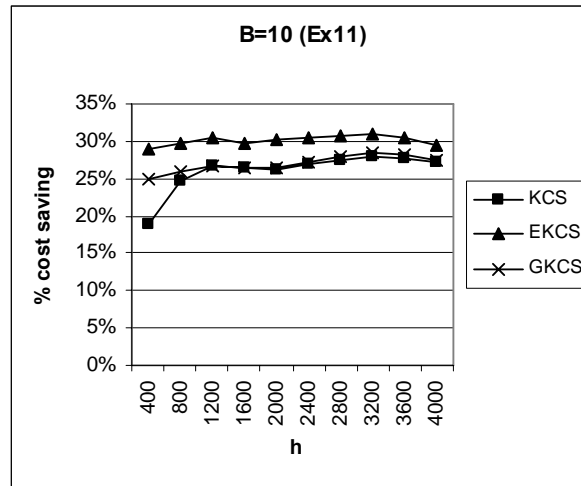


Figure 17. Per cent cost savings over BSCS as h changes

Figure 18 depicts the per cent cost savings over BSCS as h changes and when we have $b=1600$, $\alpha=0.01$, $c_R=10$, $c_M=30$, $c_D=2.5$ and $B=1$. High kanban sizes result in high holding costs, and thus KCS gets worse than BSCS. GKCS is better than KCS. As a consequence of its multi-parameter structure, EKCS results in lower inventory holding costs. When the buffer size of cores is small, KCS performs the worst as h increases, whereas the other three policies perform almost the same, especially for small values of out-of-pocket holding cost. EKCS gives still better results as a consequence of its multi-parameter structure.

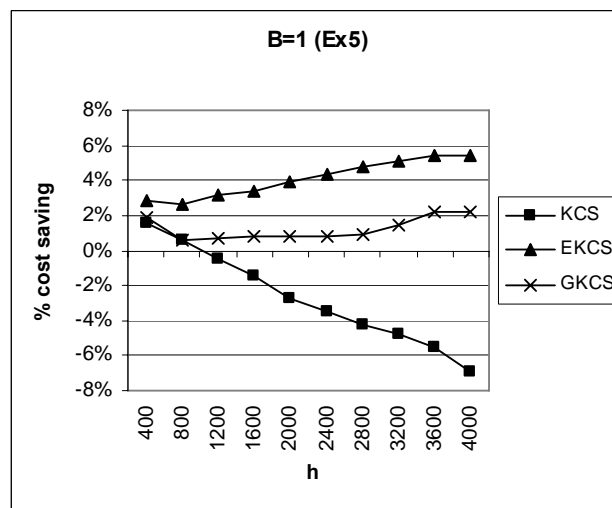


Figure 18. Per cent cost savings over BSCS as h changes

5.3. Change in Profitability

Unit production costs for remanufacturing is determined with respect to the “profitability” of remanufacturing a core as $(c_M+c_D-c_R)/c_M$. Figure 19 depicts the per cent cost savings over BSCS as c_R changes and when we have $b=3200$, $h=400$, $\alpha=0.01$, $c_M=30$, $c_D=-2.5$ and $B=10$. That WIP values have no bound in the BSCS, is the reason for BSCS performing worse than KCS, because the related holding costs increase accordingly. In the EKCS, since the buffer size of cores is high, the incoming demand is mainly satisfied from remanufactured items considering the routing probabilities. This is valid for GKCS also. However, using dedicated kanbans in the EKCS makes itself still a more preferable system in comparison with GKCS as a consequence of shorter backlog queues and work-in-processes, and their related lower costs (Figure 19).

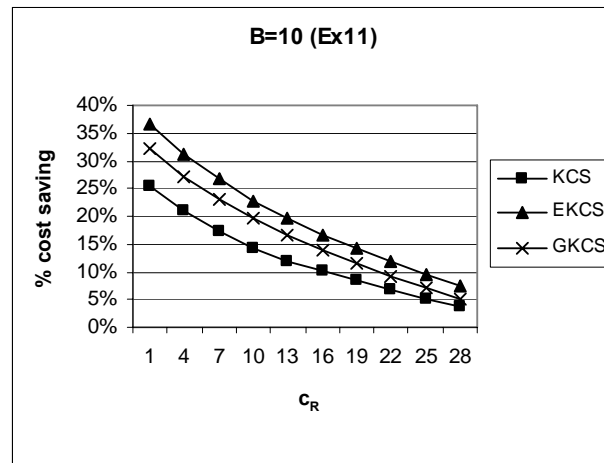


Figure 19. Per cent cost savings over BSCS as c_R changes

Figure 20 demonstrates the per cent cost savings over BSCS as c_R changes and when $b=1600$, $h=800$, $\alpha=0.01$, $c_M=30$, $c_D=2.5$ and $B=1$. The higher c_R , viz. the less the profitability is, the higher the total average costs become (Figure 20). Not only the decline in profitability, but also the small buffer size of cores forces all control mechanisms to remanufacture less as c_R increases. Since the incoming demand triggers the whole system immediately, BSCS uses

less amount of base stocks compared to KCS's kanban sizes. So, KCS results in higher holding inventory costs than BSCS. Thus, KCS becomes the worst one in this case. GKCS compensates the problem with its two-parameter structure by increasing one parameter and decreasing the other. EKCS's multi-parameter structure provides the best results among all others.

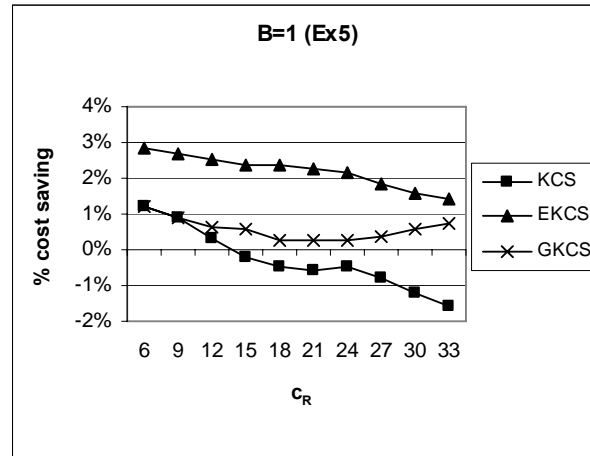


Figure 20. Per cent cost savings over BSCS as c_R changes

5.4. Change in Return Arrival Rate

We use the case where $b = 1600$, $h = 800$, $\alpha = 0.01$, $c_R = 10$, $c_M = 30$ and $c_D = 2.5$ and $B=5$. The demand arrival rate is set to be 400. As the return arrival rate decreases, the total average cost of each system increases, because the lower the return arrival rate is, the more the systems are forced providing finished products from manufacturing which is expensive. As the return arrival rate γ decreases, the optimal static routing probabilities also decrease. Hence, $(1-r)$ increases to satisfy the demand as expected. The advantage of EKCS, here, is, that it decreases base stock level to provide compensation. EKCS is the best control policy since it has the most parameters among all. However, BSCS is the worst policy (Figure 21).

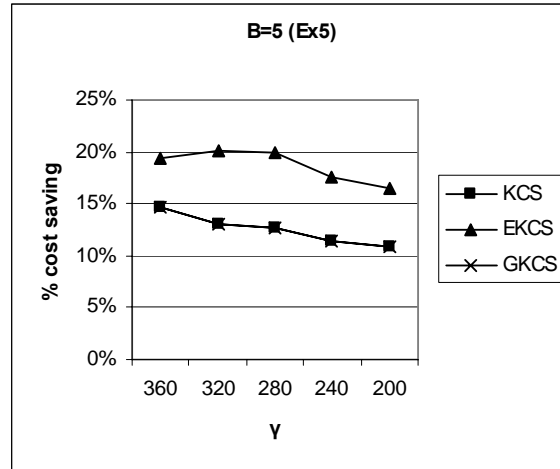


Figure 21. Per cent cost savings over BSCS as γ changes

5.5. Change in Demand Arrival Rate

The buffer size of cores B is set equal to five, whereas the demand arrival rate varies from 400 down to 300. The return arrival rate is set to be 360. The service rates of both processes are as mentioned before.

In the KCS, as the demand arrival rate decreases, the holding cost associated with both non-serviceable items and finished products decreases as long as $\gamma \leq \lambda_D$, and then increases for $\gamma > \lambda_D$. Meanwhile, the backorder cost increases as long as $\gamma \leq \lambda_D$, and then decreases for $\gamma > \lambda_D$. Here, the disposal cost decreases as long as $\gamma \leq \lambda_D$, and then increases for $\gamma > \lambda_D$. Remanufacturing cost increases as long as $\gamma \leq \lambda_D$, and then decreases for $\gamma > \lambda_D$. Manufacturing cost decreases as λ_D decreases, and these changes lead to a total decrease in overall cost. The slower the demands arrive at the system compared to the return flow, the more similar behavior KCS, GKCS and EKCS have. The per cent cost savings increase as long as $\gamma \leq \lambda_D$, and decrease for $\gamma > \lambda_D$. BSCS should be avoided for every arrival rate value. The increase in the GKCS happens quicker than in the EKCS. GKCS seems to be at least as good as KCS. For $\gamma \leq \lambda_D$, one should one should prefer EKCS, whereas for $\gamma > \lambda_D$ it is reasonable to use a less complicated system such as KCS (Figure 22).

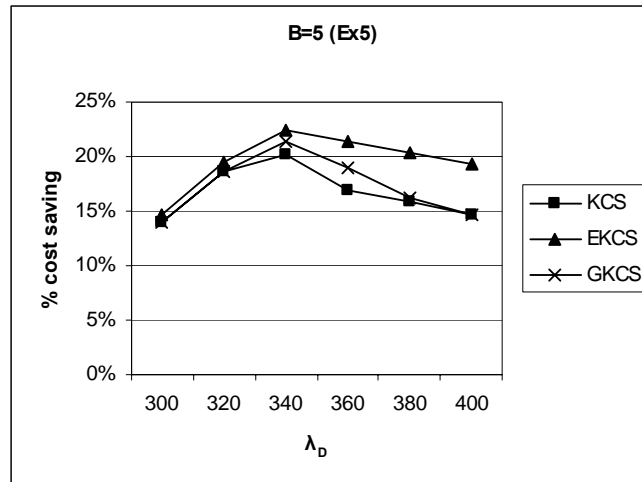


Figure 22. Per cent cost savings over BSCS as λ_D changes

6. Conclusions

The objective of this study is to compare the cost effectiveness of pull control policies in hybrid manufacturing / remanufacturing systems. To this end, first we adapted the most common four pull control policies in the literature to the hybrid production system we proposed, viz. KCS, BSCS, EKCS and GKCS. Here, we analyzed these systems using a decomposition approximation. The adapted analytical methods for each control policy provided us performance measures required for the evaluation of these systems, in terms of their total average costs. These performance measures are, for instance, the average queue length of cores, the average finished product queue, the average work-in-process queue of both remanufacturing and manufacturing processes and the average backorder queue.

After obtaining necessary performance measures, we defined a cost function that consists of holding, backorder, variable production and disposal costs. Holding cost for cores, work-in-processes and finished products, and backorder cost are calculated by multiplying their cost rates with their respective average queue lengths. Disposal cost is obtained by multiplying the unit backorder cost with the disposal rate, and, production costs are calculated by multiplying unit production costs with their related expected throughputs. Summation of these components provided us with the cost function that we want to optimize. Then we

searched for local minima for the total average cost with respect to the kanban sizes, base stock levels and static routing probabilities.

After defining the cost functions for each control policy, we compared them using an experiment set generated following the Taguchi method. Then we selected some experiments in order to perform sensitivity analysis. We analyzed, how independent changes in backorder cost, holding cost, remanufacturing and manufacturing cost and disposal cost affect the performances of each system.

We observed from the numerical results, that the control policy which provides the best performance in terms of cost is EKCS in all cases considered, because the more complicated the system is getting, and the more parameters it utilizes to control the hybrid production system, the lower total cost it provides. GKCS follows it with a small margin, as the second best control policy. This fact is a consequence of the higher flexibility related to complicated systems that results in the capability of balancing the disposal and, holding and production costs with the backorder cost more effectively. This result is parallel to the results coming from the analysis of pull type controlled ordinary production systems. However, in determining the factors on the superiority of complex pull control systems for hybrid production systems, there are some distinct cases that are different than those of ordinary production systems.

While in traditional production systems the backorder cost is the main motive behind generating complex control mechanisms, in hybrid production systems, two additional factors play an important role. Here, as marginal profitability of remanufacturing increases, complex control mechanisms perform significantly better. Similarly an increase in the return rate of cores while it stays below the demand rate renders better results with complex mechanisms. In our experiments we observed cost saving around 25 per cent. Consequently, if the remanufacturing is not profitable and there is not enough supply of returns for

remanufacturing, viz. when the return rate is smaller than the demand rate, or if the remanufacturing is not profitable and when there is excess supply of returns, there is no need to implement complex control mechanisms to the production system.

Another dimension of the problem is the effect of the buffer size on the superiority of the complicated control policies. We observed that, in hybrid production systems with large buffers size of cores, complicated control policies give higher cost savings than those of the systems with smaller buffer sizes. For instance, in cases where the capacity of the core buffer and marginal profit of remanufacturing is high, the cost effectiveness of EKCS and GKCS moves away from that of KCS and BSCS. However, for small capacity of core buffer size, the cost saving of complex control policies is not that significant, even if the marginal profit of remanufacturing is high compared to the manufacturing of raw materials.

When complex control policies do not provide high cost improvements, preferring simple control policies is a more appropriate decision because of their simplicity in factory applications. However, when we are faced with the decision of selecting one of the simple control policies viz. KCS and BSCS, some interesting factors that are unique to the special nature of hybrid production systems become apparent. In fact, as the marginal profitability of remanufacturing increases, as backorder cost increases, holding cost decreases, and as return rate increase, KCS provides a cost saving of about 1-2%. However, as the profitability decreases, as backorder cost decreases and as holding cost increases, KCS is significantly worse than BSCS. These observations are slightly different than those of ordinary systems. Also, we observed that the buffer size of cores plays an important role in deciding which policy is better. In experimental findings, it is apparent that when the buffer of cores has high capacity, KCS is better than BSCS, whereas when the buffer of cores has small capacity; BSCS can perform significantly better than KCS.

In conclusion, we see that pull type control of hybrid production systems with complex control policies displays a better performance than that of simple policies, as in the case of ordinary production system. However, it is also seen that the factors, which make this difference significant, can be different than those of ordinary production systems because of the special characteristics of hybrid production systems. So, there are some interesting cases where unique characteristics of remanufacturing yield unique managerial decisions for the selection problem of pull type control mechanisms, and our research gives a roadmap for those decision makers who are trying to cope with the uncertainties of remanufacturing by using pull type production control mechanisms.

References

Baynat, B., Y. Dallery, M. Di Mascolo, and Y. Frein, 2001, "A multi-class approximation technique for the analysis of kanban-like control systems", *International Journal of Production Research*, Vol. 39, No. 2, pp. 307-328.

Bruell, S. C. and G. Balbo, 1980, *Computational Algorithms for Closed Queuing Networks*, Elsevier North Holland Inc.

Dallery, Y., and G. Liberopoulos, 2000, "Extended kanban control system: combining kanban and base stock", *IIE Transactions*, Vol. 32, pp. 369-386.

Duri, C., Y. Frein, and M. Di Mascolo, 2000, "Comparison among three pull control policies: kanban, base stock and generalized kanban", *Annals of Operations Research*, Vol.93., pp.,41-69.

Karaesmen, F., and Y. Dallery, 2000, "A performance comparison of pull type control mechanisms for multi-stage manufacturing", *International Journal of Production Economics*, Vol. 68, pp. 59-71.

Korugan A. and S. M. Gupta, 2001, "Pull Type Control Mechanisms for Single Stage Hybrid Manufacturing Systems," Laboratory for Responsible Manufacturing, Northeastern University, Boston LRM-001.

Liberopoulos, G., and Y. Dallery, 2000, "A unified framework for pull control mechanisms in multi-stage manufacturing systems", *Annals of Operations Research*, Vol.93, pp.325-355.

Teunter, R., E. van der Laan, and K. Inderfurth, 2000, "How to set the holding cost rates in average cost inventory model with reverse logistics?", *Omega*, Vol. 28, pp.409–415.

van der Laan, E. V., M.Salomon, R. Dekker, and L. Wassenhove, 1999, "Inventory control in hybrid systems with remanufacturing", *Management Science*, Vol. 45, No. 5, pp. 733–747.